

Example of doing SUSY particle studies with the LCD environment

(Model Line B: -ino Mass Reconstruction)

13 July 2001 @ Snowmass 2001

- H. Neal

The Study:

For Model Line B

$M_3 = M_2$ (at the "GUT" scale) varies as the dimensionful
parameter,

bino mass parameter $M_1 = 1.6 M_2$ (at the GUT scale),

$m_0 = 0.5 M_2$,

$A_0 = 0$,

$\tan\beta = 10$,

$\mu > 0$.

[non-universal SUGRA with NUSUG1 option in ISAJET 7.51]

characteristics:

LSP is bino like neutralino - -ino mass diff. very small

reconstruct mass of the chargino and neutralino in

$$\chi^+ \chi^- \rightarrow W^+ \chi^0 W^- \chi^0 \rightarrow qq\chi^0 + l\nu\chi^0$$

events using the end-points of the energy or boost distribution
of the W reconstructed from the qq jets

two body kinematics of the $\chi^+ \rightarrow W^+ \chi^0$ decay implies a box shaped
distribution of the W energy

with edges at $E'_W = \gamma E'_W \pm \beta \gamma p'_W$

$E'_W = \frac{M_{\chi^+}^2 - M_{\chi^0}^2 + M_W^2}{2M_{\chi^+}}$ is the W energy, and p'_W the momentum in the χ^+ rest

frame and γ and β are of the χ^\pm .

(large resolution effects especially important for small mass differences!)

Getting from the Initial to Final State

*generally followed recommendations given on distributed CD
generated STDHEP files using isajet and pythia
started with isajet source and just built it (to avoid installing JAS)*

- simulated/reconstructed events using routine
based on the provided
Software/RootApps/HtmlDocs/Examples3.2/root_GoFastMC.C
routine

- Event selection routines combined with GoFastMC.C
+
small modifications to write out ROOT classes

.....

NoteLselection including jet-finding, jet flow routines etc... all run under
ROOT

(note: also ran standalone (C++) and verified results)

- samples analyzed and combined by storing results in ROOT histograms
then fitting and combining them using Cint routines
-

Tools:

- Linux system
 - ISAJET 7.51 - for signal generation
 - LCDROOT V3.2 fast Monte Carlo Signal Simulation
 - Pythia for WW and two-photon background simulations
- Root 3.00/Cint/LCDROOT for event selection, distribution generation,
event displays

An ISAJET Job

<p><u>A shell script:</u></p> <pre>ij751 <</EOF isadecay.dat nusug.input 10000 nusug.std /EOF</pre>	<p><u>The input file (nusug.input):</u></p> <pre>SAMPLE POLARIZED E+E- NUSUGRA JOB, MODEL LINE B, MHALF=200 500.,5000,10,10/ E+E- SUGRA 100,200,0,10,+1/ NUSUG1 320,200,200/ TMASS 175,-1,1/ EPOL -.9,0./ JETTYPE1 'W1SS-','W1SS+' JETTYPE2 'W1SS+','W1SS-' NTRIES 10000/ EEBEAM 400.,500.,.11,.11/ END STOP</pre>

Model line B points chosen:

m_0 (GeV)	M_1 (GeV)	$M_{2,3}$ (GeV)	A_0	$\tan\beta$	μ	chargino (GeV)	neutalino (GeV)
150	480	300	0	10	>0	215.8	182.1
100	320	200	0	10	>0	133.7	117.7

Information provided by ISAJET:

```
*****
*
*   ISAJET      V7.51    10-MAY-2000 20:15:21
*
*****

SAMPLE POLARIZED E+E- NUSUGRA JOB, MODEL LINE B, MHALF=300

NUMBER OF JETS TO BE GENERATED PER EVENT  2

P          ON P          AT COM ENERGY      0.5000E+03

NUMBER OF E+E-      EVENTS TO BE GENERATED      100

PRINT A MAXIMUM OF      10 EVENTS SKIPPING      10 EVENTS AT A TIME

RUN ID      010709      142655
JET NO.  2

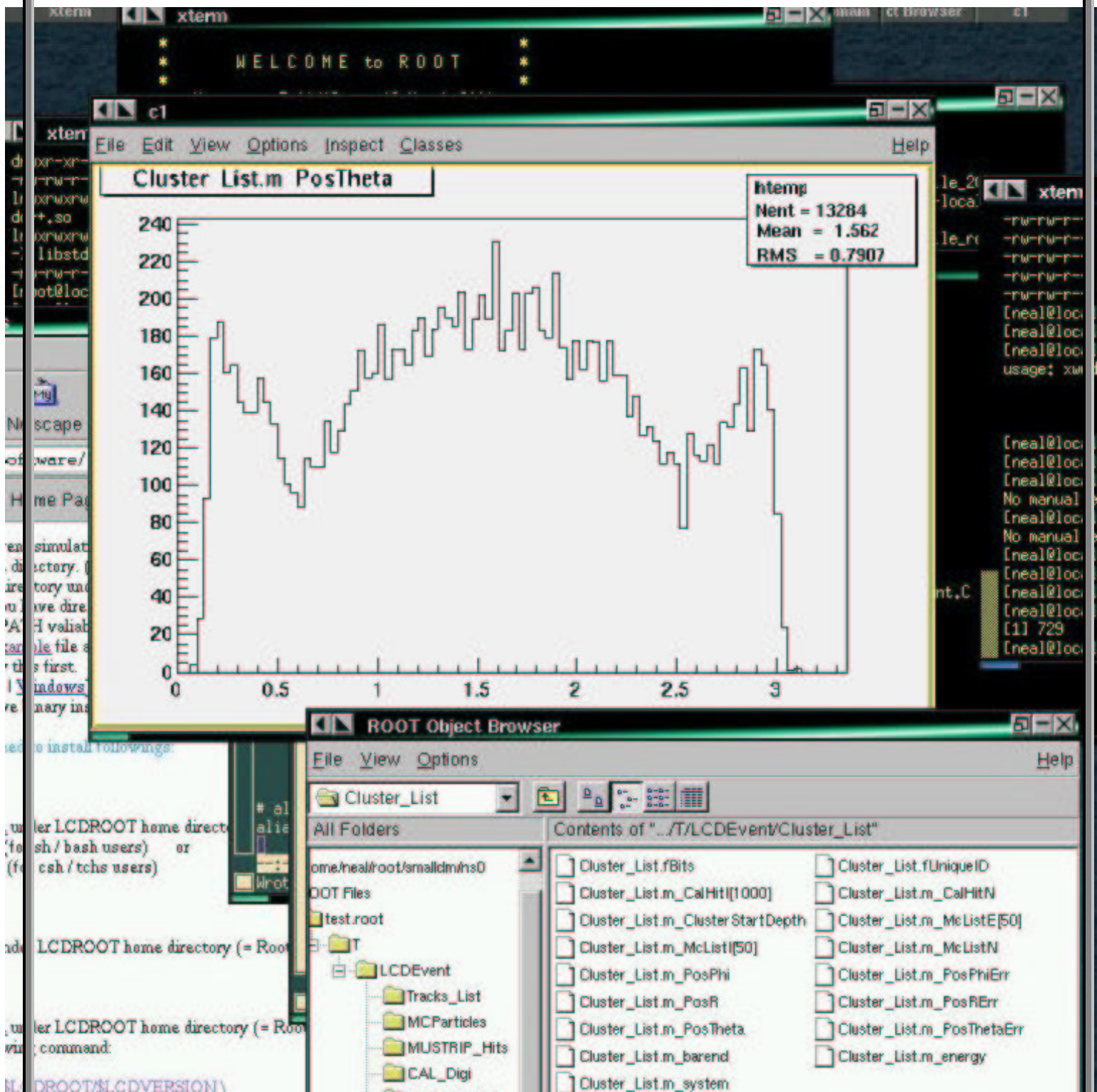
HEAVY QUARK MASSES =  175.00    -1.00    1.00

MSSM MASSES (WITHOUT SIGNS):
M(NUEL ) =  251.509      M(EL- ) =  263.655      M(NUML ) =  251.509
M(MUL- ) =  263.655      M(NUTL ) =  250.499      M(TAU1-) =  222.833
M(ER- ) =  231.259      M(MUR- ) =  231.259      M(TAU2-) =  267.957
M(GLSS ) =  708.501      M(Z1SS ) =  182.132      M(Z2SS ) =  217.581
M(Z3SS ) =  403.634      M(Z4SS ) =  423.670      M(W1SS+) =  215.801
M(W2SS+) =  422.818      M(HL0 ) =  114.700      M(HH0 ) =  465.110
M(HA0 ) =  464.265      M(H+ ) =  471.582

DETERMINED FROM SUGRA INPUT:
M_0      =  150.000      M_(1/2) =  300.000      A_0 =  0.000
TAN(BETA) =  10.0000      SGN(MU) =  1.0
```

(note: non essential lines removed)

Root permitted immediate ability to check simulated events
(just open the root file produced by the FastMC and use
Tbrowser b command to explore variables with a few clicks)



Note: In case Root cannot find the classes use a command like ->
gSystem->Load("/usr/lib/RootApps/V3.2/lib/Linux/libLCDEvent.so");

Event Simulation was Simple

Used the **GoFastMC.C** example from the CD

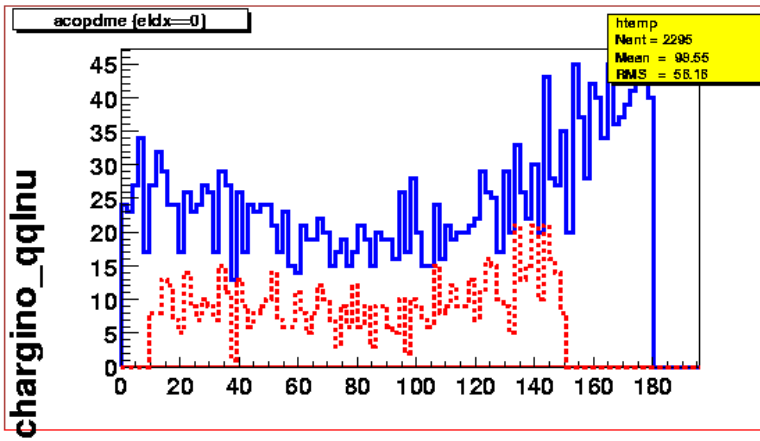
Detector Model easily switched by changing two lines

```
// Open detector parameter file  
Char_t* geomFile =  
"Silicon.par"; //Silicon  
// "Large.par"; // Large  
// "Small.par"; // Small  
  
// Tracking smearing parameter file  
Char_t* smearFile =  
"lookup_silicon.nobmcon";  
// "lookup_large.nobmcon";  
// "lookup_small.nobmcon";
```

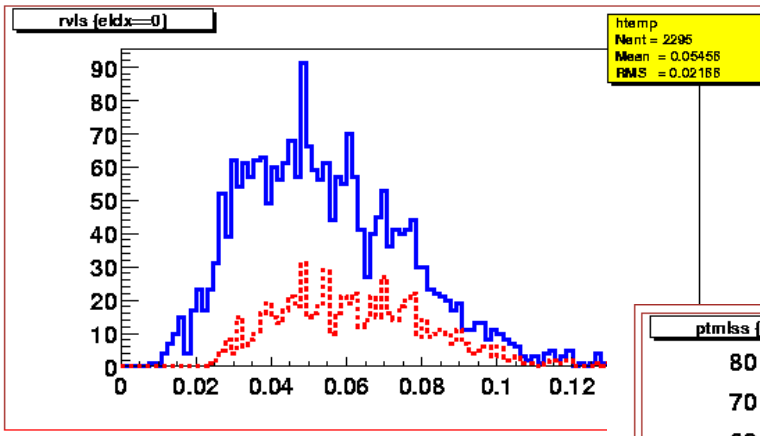
Wrote output (simulated events) to Root file
and directly analyzed using classes created by the FastMC.

Could equally as well just run analysis on the Root output.

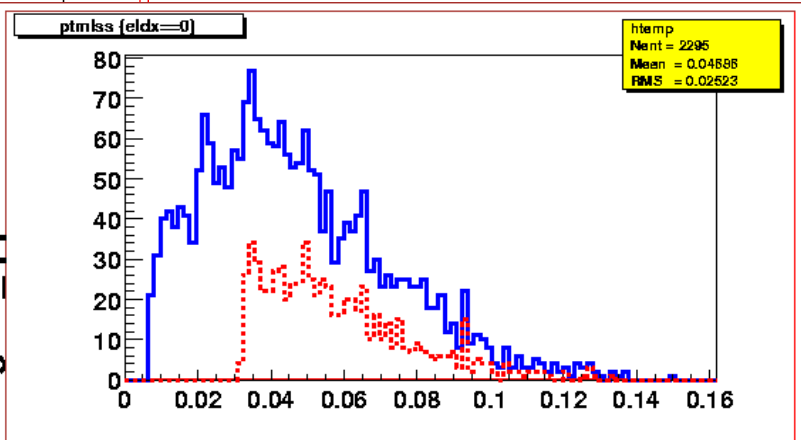
Event Variables



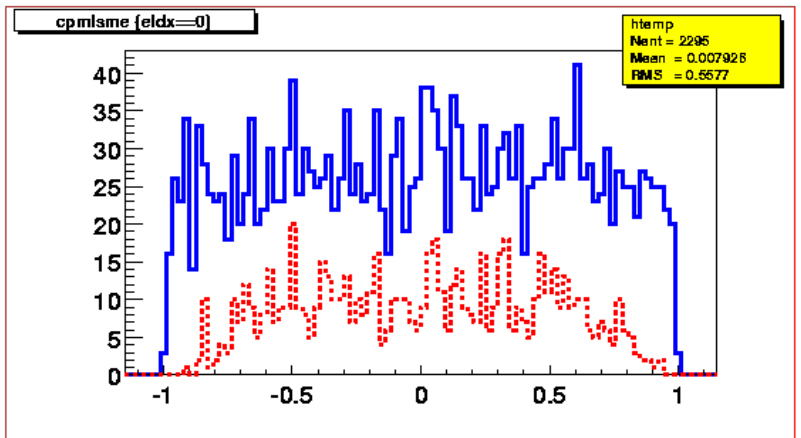
acoplanarity
and
visible energy



chargino_qqlnu

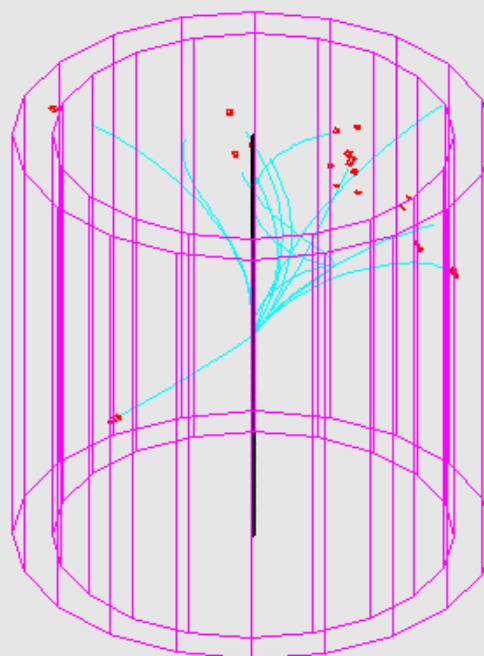
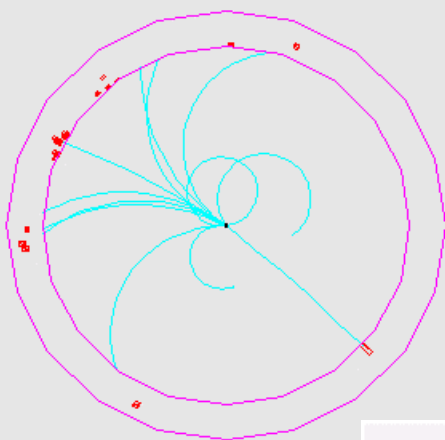


missing Pt
and
polar angle of missing
momentum

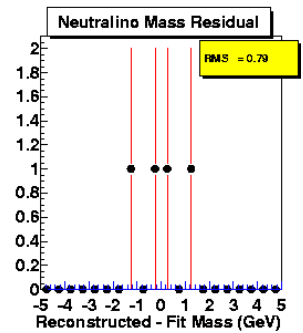
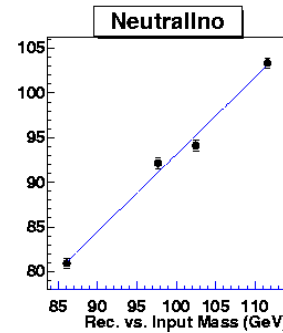
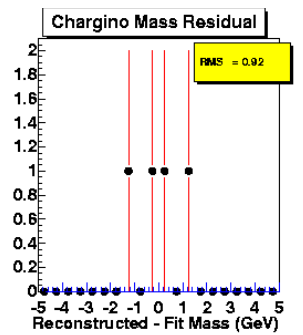
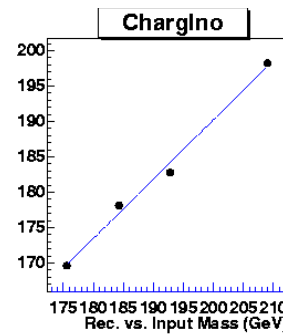
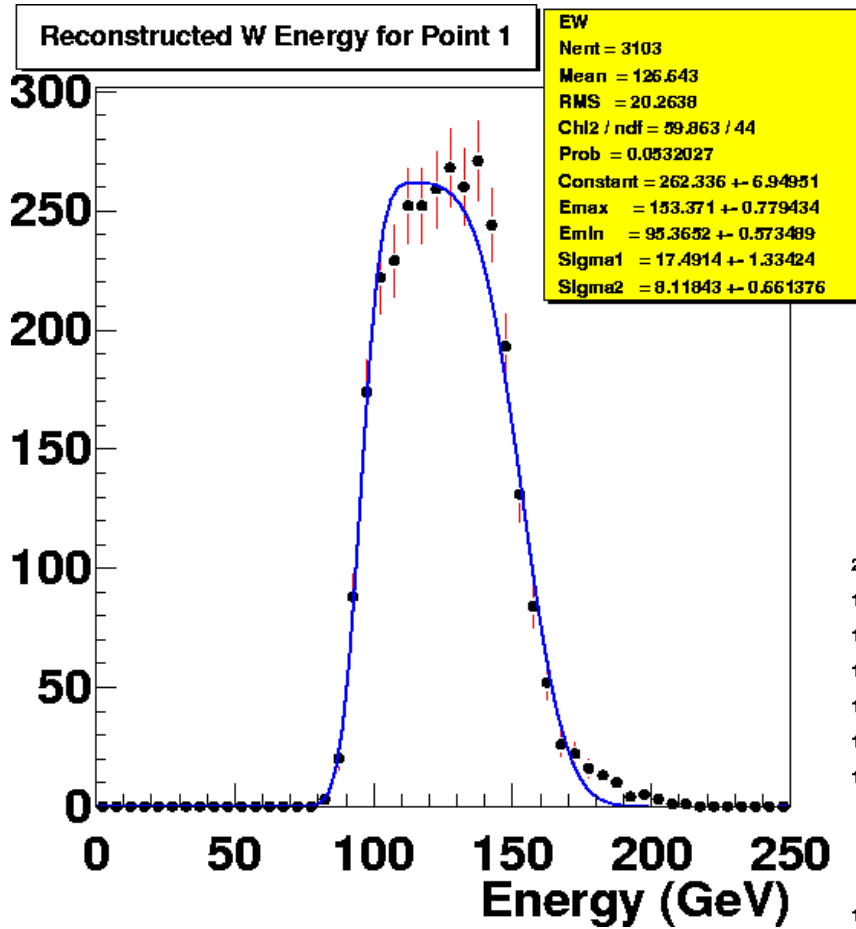


Event Displaying was Simple

Using root_EventDisp.C on the CD and the root file produced by the
FastMC

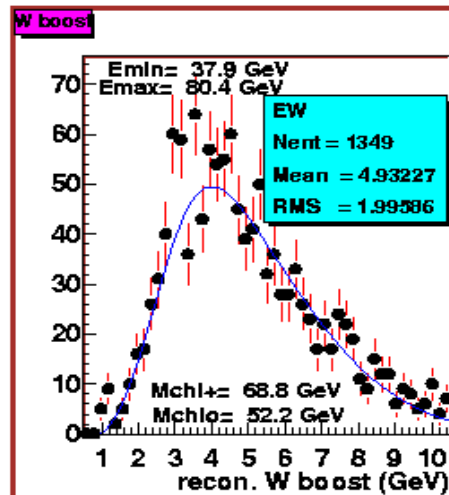
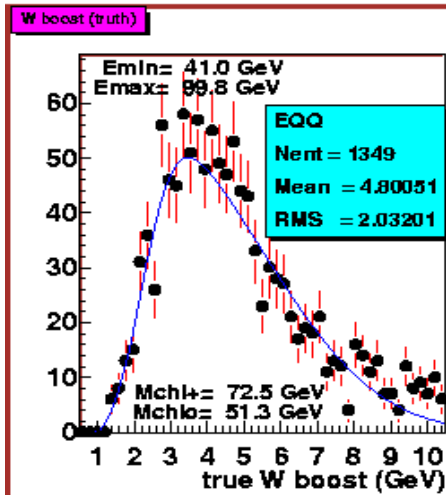
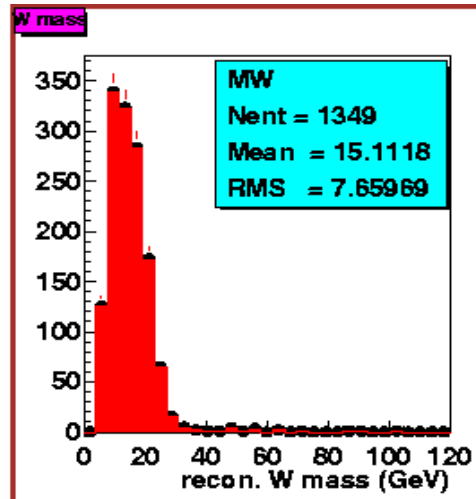
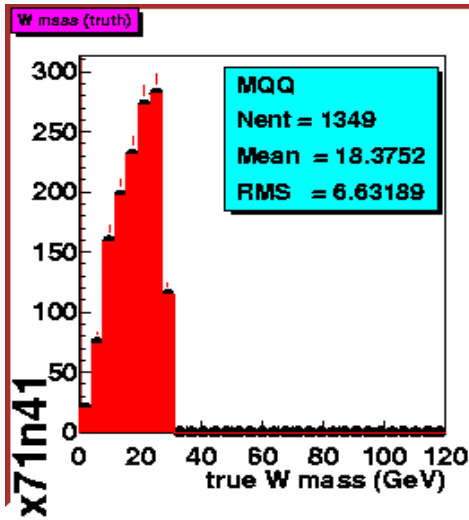


An Easy Case (large mass difference):



Set	m_{χ^\pm}	m_{χ^\pm}	m_o	$m_{1/2}$	A_o	$\tan\beta$	sign μ
1	175.6	86.1	400.	200.	0.	2.0	-

life not so easy for small mass differences \rightarrow



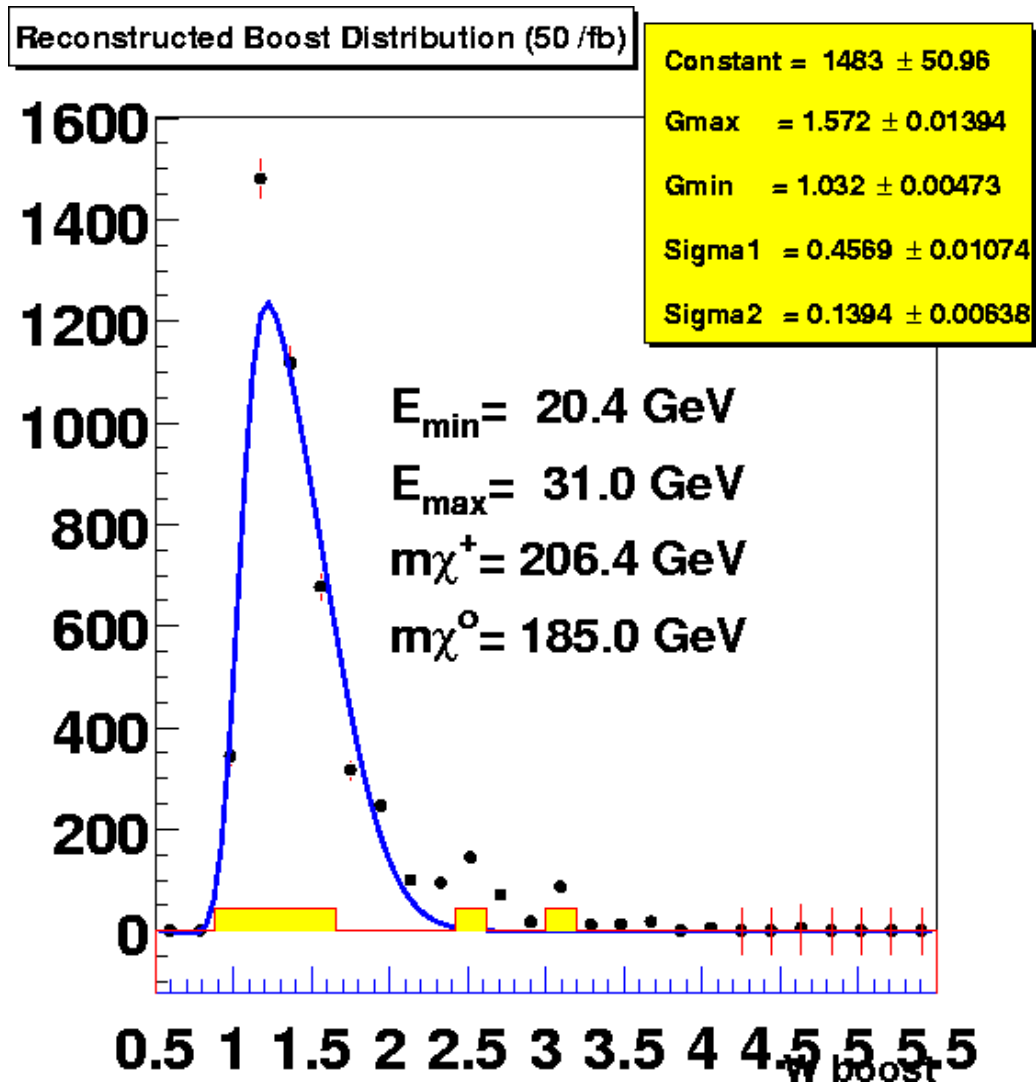
Analysis reformulated in terms of the minimum and maximum observed **boost** which results in many of the detector uncertainties canceling out

$$m_{\chi^\pm} = \sqrt{\frac{(1 + \gamma_{\max} * \gamma_{\min}) - \sqrt{(1 - \gamma_{\max})^2 \times (1 - \gamma_{\min})^2}}{2(\gamma_{\max} + \gamma_{\min})^2 / E_{COM}^2}}$$

Set	m_{χ^\pm}	m_{χ^\pm}	m_o	$m_{1/2}$	A_o	$\tan\beta$	sign μ
1	74.8	43.1	150	120	350	35	+
2	70.7	40.8	140	115	350	35	+
3	58.4	33.8	150	100	350	35	+
4	37.0	20.8	500	60	350	35	+

Some Results Already...

(for Model Line B points)



Expected: $m_{\chi^+} = 215 \text{ GeV}$ and $m_{\chi^0} = 182 \text{ GeV}$

Can be calibrated

(from nothing to this point took several days
using the LCD software)

As Easy As ...

(from the LCD CD)

- **ROOT** : We need **version 3.00.06**.
This CD has tar.gz files for [[Linux \(Redhat6.1 glibc2.1 egcs1.1.2\)](#) | [Windows](#)], or [download from ROOT page](#).
- Just unpack them where you like. The directory you install becomes ROOTSYS.
- **LCDROOT** : This is the main program for event simulation/analysis. This CD has LCDROOT V3.2.
 - **Setup 1** Make your LCDROOT home directory. (= RootApps or so.)
 - **Setup 2** Make your LCDVERSION directory under the above directory. (=V3.2)
Note:: After doing Setup 1 & 2, you have directory tree like :
RootApps/V3.2
 - **Setup 3** Setting the ROOTSYS and PATH variables for [[sh/bash](#) | [csh/tcsh](#) | [Windows](#)] users.
 - **Setup 4** Copy the following [rootrc example](#) file as .rootrc in your home directory.
 - **Binary installation** ... You should try this first.
Copy and unpack tar.gz file [[Linux](#) | [Windows](#)] under V3.2 directory.
- **LCDROOT Event classes structure (V3.2)**
- **Example Event Generation/Analysis Files** (These files are in the Examples directory)

Summary Report

Thanks to the efforts of many people

Peter Armstrong Joanne R. Bogart Gary Bower Toby Burnett Ron E Cassell Richard Dubois	Dan Flath Ray Frey David Gerdes Jeff Gronberg Jeremy Martin Hill Masako Iwasaki Tony Johnson	Michael Peskin Michael Ronan Bruce Schumm Rick Van Kooten Tony Waite Wolfgang Walkowiak Haijun Yang
--	--	---

the LCD simulation is a viable tool for
performing future linear collider studies
on reasonable time scales

The essential tools are present as well as relevant
examples that make getting started fairly easy.

The Root based system is comfortable for those of
us already using Root

The JAVA system is supposed to be at least as
easy to use.

In order to make comparisons, a TESLA detector
model for the LCD simulation should be added.

Experts on Call:

Norman Graf (ngraf@slac.stanford.edu) - Event Generation, Analysis.
Tony Johnson (tonyj@slac.stanford.edu) - JAS, LCD utilities and Infrastructure, WIRED.
Gary Bower (grb@slac.stanford.edu) - Jet Finding, Event Generators, Analysis.
Wolfgang Walkowiak (walkowia@slac.stanford.edu) - MCFast, ZVTop (Vertexing), WIRED.
Mike Ronan (ronan@lbl.gov) - Track Reconstruction, JAS, Analysis.
Masako Iwasaki(masako@slac.stanford.edu) - pandora-pythia, LCDROOT, Analysis
Toshi Abe(toshi@slac.stanford.edu) - LCDROOT, Analysis
Bob Wilson (wilson@lamar.colostate.edu) - Particle ID, JAS
Ron Cassell (cassell@slac.stanford.edu) - GISMO, LCD Utilities